Module 6 | Attacking the Stack

Introduction

Binary exploitation! We're going to build a vulnerable program and exploit it!

We've been working toward this point throughout the course. At this point, you should be familiar with binary exploitation and when it's used, and somewhat familiar with the basic constructs we'll take advantage of. This module's highly technical; we'll start using GDB to exploit programs, start generating binary injection payloads, analyzing core files, and really looking closely at how stack exploitation works. By the end of this module, we will have exploited a sample program via a buffer overflow, and will have executed arbitrary code that we've injected.

Note: Some installations of Ubuntu LInux now use ASCII Armoring (a defensive technique that places important functions like the LIBC functions in addresses with a terminating NULL value, which makes direct address injection impossible). You may want to roll back to an older version of Linux if this is the case on your installation.

Learning Objectives

The objectives of this module, like most of the modules in this course, are to teach you how to design and execute malware as well as defend systems against it. We'll be addressing these topics from a much different perspective in this module than we have so far. This module has one homework assignment in which you build your own exploitable program, and then exploit it.
The other objective is to learn how to compromise a program via programming flaws.

All homework and quizzes should to be submitted from this module by the end of the sixth week. You'll use GCC, GDB, and python in this assignment. In the assignment associated with this module you'll create an exploitable program and then feed the program corrupted input to call an uncalled function.

Required Instructional Materials

Follow the videos in your Ubuntu binary exploitation environment. Make sure you take notes when appropriate, and try to ensure that you understand exactly what's going on at all times.

Summary

We're going to learn binary exploitation. We've been working toward this point through the term, and we're finally here! This module is more difficult than the previous, so make sure you've scheduled enough time to review and experiment with the material and complete the homework. Be patient! Binary exploitation work can be frustrating as it's so detail oriented, but you'll get there!

If you have questions about some aspect of Learn, **UNM LEARN Support** is available to troubleshoot technical problems.

Contact them 24/7 at 505-277-0857, 1-877-688-8817 or use the "Create a Support Ticket" link on the left Course Menu.